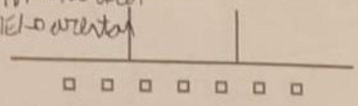
Subgrafos dentro de um grafo completo de N vértices = 2^n

|F| → faces internas
|V| → vértices
|E| → arestas



Grafos Completos (K_n) → todo vértice ligado com todo vértice   arestas = $N*(N-1)/2$ $|F|+|V|=|E|+2$

Grafo Regular → todo vértice possui grau   Handshaking Lemma → o nº de vértices de grau ímpar deve sempre ser par

Soma dos graus dos vértices deve sempre ser par

Caminho → sequência de vértices com uma aresta entre cada dois vértices → simples: sem repetição de vértice → ciclo: termina no vértice que começou

Ciclo Euleriano → contém todas arestas do grafo apenas uma vez (conexo e todo vértice com grau par) ↳ caminho euleriano, 2 vértices têm grau ímpar

Ciclo Hamiltoniano → contém todos os vértices do grafo apenas uma vez

Grafo Bipartido → divide os vértices em 2 conjuntos, em que um vértice não se liga com outros de seu conjunto (só do outro)

Conectividade → grafo conexo é aquele que, a partir de qualquer vértice, é possível chegar em qualquer outro

Componentes Conexas → subgrafos conexos dentro do grafo desconexo (maximais)

Conectividade em vértices → menor número de vértices que removidos tornam o grafo desconexo (menor corte de vértices)
↳ K-conexo → conectividade é $\geq K$ → biconexo = 2 vértices → componente biconexa: formada pela remoção dos vértices

Subgrafo induzido → para um conjunto de vértices que forma um subgrafo, possui todas arestas deles

Árvores → grafos não dirigidos, acíclicos e conexos → $A(V, a)$, $a = V-1$
↳ possui apenas 1 caminho entre cada par de vértices   ↳ se adicionar uma aresta, cria um ciclo
↳ árvore geradora → subgrafo que contém todos os vértices originais, mantém a estrutura de árvore, mas removendo as arestas que geram ciclos → algoritmos de Prim e Kruskal

Prim → Dijkstra que não soma a distância, mede apenas pelo menor valor de aresta que o conecta

Kruskal → adiciona a aresta mais barata e seus vértices, sem se preocupar em manter conexo (apenas evitando o ciclo)

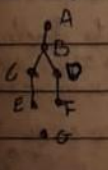
Minimal → subgrafo que atende a uma condição e não contém nenhum outro subgrafo que atenda tais condições

Maximal → subgrafo que atende à uma condição e não está contido em nenhum outro subgrafo que atenda tais condições

→ algoritmo de Moore

BFS → breadth-first search → busca em amplitude → adiciona por nível os adjacentes

DFS → vetor bool de visitados construído de forma recursiva através de um caminho. Depth-first search (profundidade)



BFS → A=0 B=1 C=2 D=2 E=3 F=3 G=-1. {A, B, C, D, E, F}

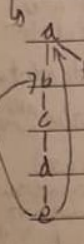
DFS → conjunto dos visitados, na ordem de visita. {A, B, C, E, D, F}. G=0, o resto=1

Encontrar saída de labirinto

- Tremaux → saída de labirinto, marca arestas percorridas, segue até não ter arestas sem marcação. Retorna pelas arestas ~~[illegible]~~ marcadas até um vértice que possui arestas não marcadas. Continua até o vértice destino.
- Hopcroft-Tarjan → ao invés de marcar as arestas, enumera os vértices e marca o pai de cada vértice. Assim, quando é preciso voltar, se usa o vetor com o pai.

Jandaia

árvore de profundidade



lowpoint: maior ponto que um vértice pode alcançar ao descer seu ramo e usar 1 aresta de retorno. Ex:

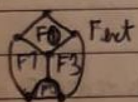
a	b	c	d	e	F
a	a	a	a	a	F

através do lowpoint:

- ponte = vértice que possui filho cujo lowpoint é ele mesmo
- articulações = raiz, caso tenha filhos > 2, ou outros vértices, caso esses tenham filhos cujos lowpoint são seus pais ou eles mesmos
- demarcadores = filhos da raiz e filhos de outros vértices cujos lowpoint é o pai ou eles mesmos

Planaridade → pode ser representado sem cruzamento das suas arestas

Face → área minimal formada por um conjunto de arestas que formam um ciclo. A região externa ao grafo é também uma face.



Fórmula de Euler → $|V|+|F|=|E|+1$ Condição: $|E| \leq 3|V|-6$

K_5 e $K_{3,3}$ são não planares, e qualquer grafo que os contenha também não é

Demoucron → verifica planaridade

1. encontra um ciclo hamiltoniano 2. pinta as arestas não usadas no ciclo

3. adiciona internamente (→ dentro do ciclo formado) as arestas não usadas

4. coloca na face interna uma aresta e passa para a externa todas que cruzam ela

5. pega as arestas agora externas e coloca como aresta interna toda aresta que lhe cruzava quando esta era interna

6. Repita o ciclo 4-5 até todas arestas terem um rótulo

Algoritmos:

Dijkstra: encontra a distância entre 2 vértices em um grafo valorado. O vértice inicial tem dist 0, os demais tem o valor da aresta que os conecta + a distância do vértice que incide, ou infinito. Inclui no conjunto o vértice de menor distância, atualiza as distâncias com as menores arestas, continua até o destino ter um valor

Bellman-Ford: dijkstra com valores negativos, atualiza o valor mesmo dos valores já no conjunto. Roda no máx. N iterações, se passar disso e houver alteração na distância, há um ciclo de soma negativa

Floyd-Warshall: programação dinâmica, mede a distância ao calcular a distância total de caminhos que passam por um outro vértice

Lowest Common Ancestor → encontra distância em uma árvore. Encontra o primeiro ancestral em comum dos vértices e soma a distância dos vértices ao ancestral

Universidade de Caxias do Sul
SIS0227A - Teoria de Grafos
Prof. Ricardo Dorneles
Avaliação da Primeira Área - Parte 1 - 03/05/22

1)(1 ponto) - Dado o grafo $V = \{A, B, C, D, E, F\}$ e $E = \{(A, C), (A, B), (B, D), (C, E), (C, D), (D, E), (D, F), (E, F)\}$ e o vértice A como ponto de partida, qual a ordem em que os vértices são visitados em uma busca em largura (0.5 pontos)? E em uma busca em profundidade (0.5 pontos)? Considere que, quando for possível escolher mais de um vértice, o de menor numeração será escolhido.

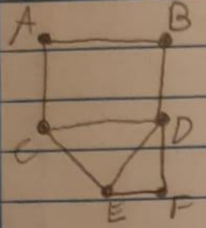
2)(2 pontos) Faça uma função que receba uma matriz de adjacências M[10][10] representando um grafo G e retorne o número de componentes conexas no grafo.

3)(2 pontos) O teorema de Ore diz que, se em um grafo, para todos os pares de vértices i e j não adjacentes, a soma dos graus de i e j é no mínimo igual ao número de vértices do grafo, então o grafo tem um ciclo hamiltoniano. Faça uma função que receba uma matriz de adjacências M[10][10] representando umg grafo G, e retorne: 1 - Se o grafo atende à condição de Ore; 0 - Em caso contrário.

Eduardo Eberhardt Pereira

5.0 ___/___/___

1)

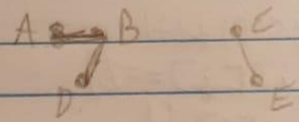


BFS -> A, B, C, D, E, F
↳ largura

DFS -> A, B, D, C, E, F
↳ profundidade

1.0

	A	B	C	D	E	F
A		1	1			
B	1			1		
C	1			1	1	
D		1	1		1	1
E			1	1		1
F				1	1	




```
void dfs(int M[10][10], int visitados, int v){
    visitados[V] = 1;
    int j;
    for(j=0; j<10; j++)
        if(visitados[j]==0 && M[V][j]==1)
            dfs(M, visitados, j);
}
```

2)

```
int func(int M[10][10]{
    int visitados[10] = {0};
    int i, j, componentes=0;

    for(i=0; i<10; i++){
        if(visitados[i]==0){
            dfs(M, visitados, i);
            componentes++;
        }
    }

    return componentes;
```

2.0

___/___/______

3)
```
int Ore(int M[10][10]){
    int i, j, graus[10]={0};
    int result=1;

    for(i=0; i<10; i++)
      for(j=0; j<10; j++)
        if(M[i][j]==1)
          graus[i]++;



    for(i=0; i<10; i++){
      for(j=0; j<10; j++){
        if( M[i][j]==0 && i!=j)
          if ((graus[i] + graus[j]) >=10){
            result=0;
            break;
          }
        if( result==0)
          break;
    }

    return result;
}
```

2.0

Universidade de Caxias do Sul
SIS0227A - Teoria de Grafos
Prof. Ricardo Dorneles
Avaliação da Primeira Área - Parte 2 - 08/05/22

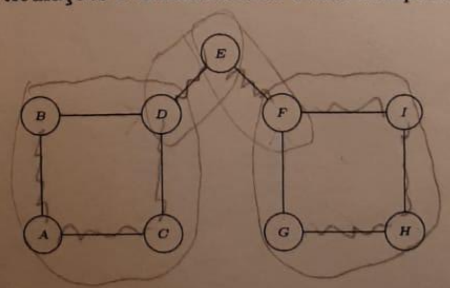
1) (1.0 ponto) Sejam G_1 e G_2 dois grafos conexos, sem vértices em comum. Seja x um vértice de G_1 e y um vértice de G_2. Seja G o grafo que se obtém de $G_1 \cup G_2$ acrescentando a aresta (x, y). É possível que o grafo G seja euleriano? E semi-euleriano (contem um caminho euleriano mas não contem um ciclo euleriano)? Em que condições?

2) (1.0 ponto)Desenhe a árvore geradora mínima do grafo representado pela matriz de adjacências a seguir:

	A	B	C	D	E	F
A		3	2		5	4
B	3		3	4		4
C	2	3		5	2	4
D		4	5		4	6
E	5		2	4		2
F	4	4	4	6	2	

3) (1 ponto) Seja T uma árvore com vértices $1, ..., n$. Suponha que os graus dos vértices $1, 2, 3, 4, 5, 6$ são $7, 6, 5, 4, 3, 2$ respectivamente e que os vértices $7, ..., n$ são folhas. Determine n.

4) (2 pontos) Decomponha o grafo a seguir em seus componentes biconexos, a partir do vértice e, utilizando o algoritmo dado. Mostre todos os passos do algoritmo (montagem da árvore de busca em profundidade e arestas de retorno, cálculo dos lowpt, identificação das pontes, identificação das articulações e demarcadores e dos componentes a cada passo).



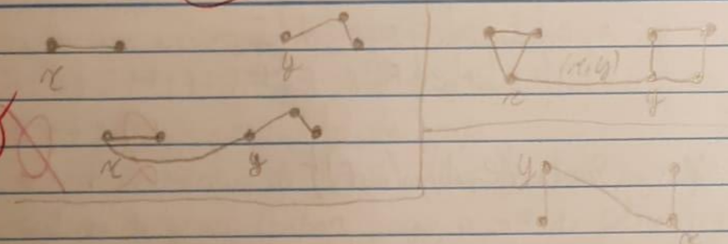
Eduardo?

5.0

___/___/___

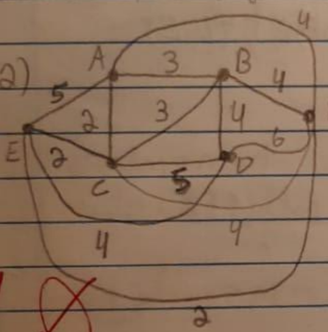
1)

1.0

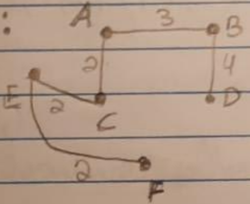


Não é possível que G seja euleriano, pois (x, y) obrigatoriamente será uma ponte, e sendo assim, não é possível formar um ciclo entre G_1 e G_2. Mas é possível formar um caminho euleriano se ambos G_1 e G_2 forem eulerianos ou caminhos eulerianos.

2)



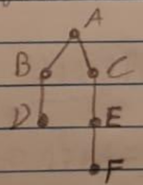
AGM:



(Prim ou Kruskal)

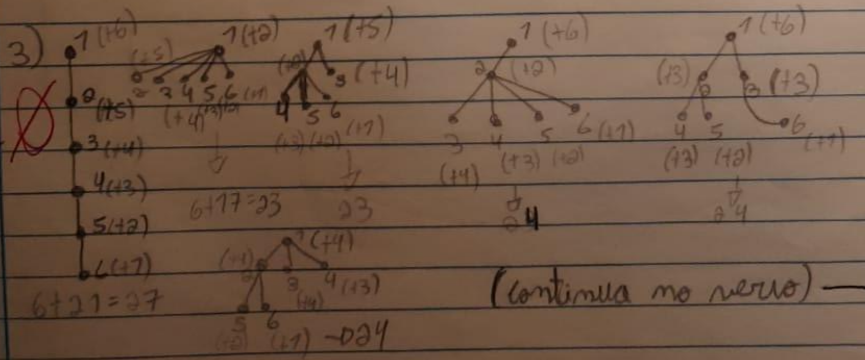
1.0

Em forma de árvore (a partir de A):

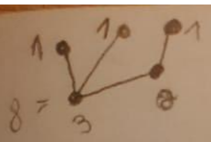


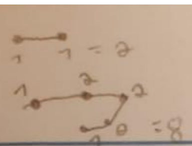
3)

1.0



(continua no verso) →





___/___/___

cont.

arestas nas árvores = $n-1$ = 5 arestas

$7+6+5+4+3+2=27$ total de graus dos 6 vértices

$27 +$ folhas = grau_total_do_grafo

o nº de vértice de grau ímpar deve ser par. Nos 6 só há 3 vértices de grau ímpar. Logo, o nº de folhas (grau 1) deve ser ímpar (p/ soma dar par). ⇒ nº de vértices é ímpar

grau_total / 2 = arestas

arestas $= n-1$

grau_total $= 2(n-1) = 2n-2$

$2n-2-27=$ folhas

arestas $= n-1$ (árvore)

arestas $= \dfrac{27 + \text{folhas}}{2}$ 2 arestas $-27=$ folhas $2n-2-27=$ folhas

$6 +$ folhas $= n$

$2(6+\text{folhas})-29=$ folhas $12+2\,\text{folhas}-29=$ folhas

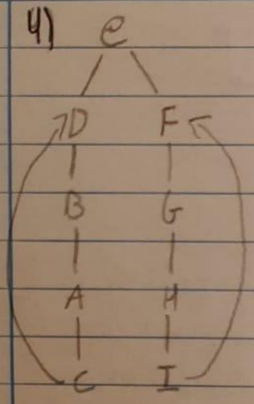
folhas $-17=0$ folhas $=17$

⇒ $17+6=$ <u>23 vértices</u>

⇒ 3

___/___/_____

4)



Lb:	A	B	C	D	e	F	G	H	I
	D	D	D	D	e	F	F	F	F

2.0

Pontes: {(e,d), (e,f)}

Articulações: {e, D, F}

Demarcadores: {D, F, B, G}

Componentes: {G, H, I, F}, {B, A, C, D}, {F, E}, {D, E}